/

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/934,659	08/22/2001	Ingo Molnar	019322-000342	7912

24239 7590 06/15/2004

MOORE & VAN ALLEN, PLLC
2200 W MAIN STREET
SUITE 800
DURHAM, NC 27705

EXAMINER	
ALI, MOHAMMAD	
ART UNIT	PAPER NUMBER
2177	10

DATE MAILED: 06/15/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No. 09/934,659	Applicant(s) MOLNAR, INGO	
	Examiner Mohammad Ali	Art Unit 2177	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *31 March 2004*.

2a)□ This action is **FINAL**. 2b)☒ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1,3 and 5-28* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1,3 and 5-28* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)□ The specification is objected to by the Examiner.

10)□ The drawing(s) filed on _____ is/are: a)□ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)□ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____.

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) □ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____.

4) □ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____.

5) □ Notice of Informal Patent Application (PTO-152)

6) □ Other: _____.

DETAILED ACTION

1. This communication is in response to the Amendment filed on March 31, 2004, Paper No. 9.

Claims 1, 3, and 5-28 are pending in this Office Action.

After a further search and a thorough examination of the present application, claims 1, 3, and 5-28 are remain rejected.

Applicant's arguments with respect to claims 1, 3, and 5-28 have been considered but are moot in view of the new ground(s) of rejection.

Applicant's representative conducted a telephone Interview on March 12, 2004 (Paper No. 8) to amend the claims to overcome the prior art of records, but happened to be that in the amendment a most significant part of the invention has been taken out from the claims, like as atomically,....

Based on the amendment the Examiner has found better prior arts to reject the existing claims.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1, 8-9, 13, and 15-28 are rejected under 35 U.S.C. 103(a) as being unpatentable over Scott T. Marcotte ('Marcotte' hereinafter), US Patent 6,449,614 in view of Pike et al. ('Pike' hereinafter), US Patent 5,623,66.

With respect to claim 1,

Marcotte discloses a method of handling a request from an application to an operating system to perform a file operation relative to a specific file (see col. 13, lines 50-60, Marcotte), the method comprising the steps of:

receiving the request to perform the file operation (see col. 16, lines 8-14, Marcotte), wherein the request to perform the file operation causes a file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte);

attempting to perform the file look-up operation without suspending task execution (see col. 14, lines 5-13, Marcotte) by retrieving a file path corresponding to the specific file from a file system namespace cache (see col. 18, lines 26-33, Marcotte);

notifying the application that the file operation could not be performed without suspending task execution so that the application can redirect the request (see col. 14, lines 5-13, Marcotte), if the file look-up operation could not be performed without suspending task execution because the file path is not stored in the file system namespace cache (see col. 18, lines 26-33, Marcotte); and

performing the file operation without suspending task execution if the file look-up operation (see col. 14, lines 5-13, Marcotte) could be performed without suspending task execution because the file path corresponding to the specific file is stored in the file system namespace cache (see col. 18, lines 26-33, Marcotte).

Marcotte does not explicitly indicate the claimed namespace for cache.

Pike discloses claimed namespace for cache (an operating system root service which provides a predefined file tree for binding to the root of the prcess's name space, see col. 3, lines 59-61, Pike).

It would have been obvious to one ordinary skill in the data processing art, at the time of the present invention to combine teachings of the cited references because the namespace for cache of Pike's teachings would have allowed Marcotte's system for multiprocess operating system which characterized in different name, as suggested by Pike at col. 3, lines 41-45. Further, Namespace for cache as taught by Pike improves to files control entity on per-process basis (see col. 3, lines 64-65, Pike).

As to claim 3,

Marcotte teaches wherein the file system namespace cache is disposed within an operating system kernel (see col. 18, lines 26-33, Marcotte).

With respect to claim 8,

Marcotte discloses a computer program product having computer program code embodied therein, the computer program code for of handling a request from an application to an operating system to perform a file operation related to a specific file (see col. 13, lines 50-60, Marcotte), the computer program code comprising:

instructions for receiving the request to perform the file operation (see col. 16, lines 8-14, Marcotte), wherein the request to perform the file operation causes a file look-up operation(see col. 18, lines 54-58, Fig. 3, Marcotte);

instructions for determining if a file path corresponding to the specific file is stored in a file system namespace cache (see col. 18, lines 26-33, Marcotte);

instructions for performing the file look-up operation without suspending task execution (see col. 14, lines 5-13, Marcotte) if the file path corresponding to the specific file is stored in the file system namespace cache (see col. 18, lines 26-33, Marcotte);

instructions performing the file operation without suspending task execution (see col. 14, lines 5-13, Marcotte) if the file look-up operation could be performed without suspending task execution because the file path corresponding to the specific file is stored in the file system namespace cache (see col. 18, lines 26-33, Marcotte); and

instructions for notifying the application that the file operation could not be performed without suspending task execution (see col. 14, lines 5-13, Marcotte) so that the application can redirect the request, if the file look--up operation could not be performed without suspending task execution because the file path is not stored in the file system namespace cache (see col. 18, lines 26-33, Marcotte).

Marcotte does not explicitly indicate the claimed namespace for cache.

Pike discloses claimed namespace for cache (an operating system root service which provides a predefined file tree for binding to the root of the prcess's name space, see col. 3, lines 59-61, Pike).

It would have been obvious to one ordinary skill in the data processing art, at the time of the present invention to combine teachings of the cited references because the namespace for cache of Pike's teachings would have allowed

Marcotte's system for multiprocess operating system which characterized in different name, as suggested by Pike at col. 3, lines 41-45. Further, Namespace for cache as taught by Pike improves to files control entity on per-process basis (see col. 3, lines 64-65, Pike).

As to claim 9,

Marcotte teaches wherein the computer program code further comprises instructions for maintaining the file system namespace cache within an operating system kernel (see col. 18, lines 50-53, Marcotte).

With respect to claim 13,

Marcotte discloses an apparatus for handling a request from an application to an operating system to perform a file operation relative to a specific file (see col. 13, lines 50-60, Marcotte), the apparatus comprising:

means for receiving the request to perform the file operation (see col. 16, lines 8-14, Marcotte) wherein the request to perform the file operation causes a file look-up operation(see col. 18, lines 54-58, Fig. 3, Marcotte);

means for determining if a file path corresponding to the specific file is stored in a file system namespace cache (see col. 18, lines 26-33, Marcotte);

means for performing the file look-up operations without suspending task execution (see col. 14, lines 5-13, Marcotte) if the file path corresponding to the specific file is stored in the file system namespace cache (see col. 18, lines 53-55, Marcotte);

means for performing the file operation without suspending task execution (see col. 14, lines 5-13, Marcotte) if the file look-up operation (see col. 18, lines

54-58, Fig. 3, Marcotte) could be performed without suspending task execution because the file path corresponding to the specific file stored in the file, system namespace cache (see col. 18, lines 26-33, Marcotte); and

means for notifying the application that the file operation was could not be performed without suspending task execution(see col. 14, lines 5-13, Marcotte) so that the application can redirect the request, if the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) could not be performed without suspending task execution because the file path is not stored in the file system namespace cache (see col. 18, lines 26-33, Marcotte).

Marcotte does not explicitly indicate the claimed namespace for cache.

Pike discloses claimed namespace for cache (an operating system root service which provides a predefined file tree for binding to the root of the prcess's name space, see col. 3, lines 59-61, Pike).

It would have been obvious to one ordinary skill in the data processing art, at the time of the present invention to combine teachings of the cited references because the namespace for cache of Pike's teachings would have allowed Marcotte's system for multiprocess operating system which characterized in different name, as suggested by Pike at col. 3, lines 41-45. Further, Namespace for cache as taught by Pike improves to files control entity on per-process basis (see col. 3, lines 64-65, Pike).

With respect to claim 15,

Marcotte discloses an operating system (see col. 13, lines 50-60, Marcotte) comprising:

a file system including; a file system namespace (see col. 18, lines 26-33, Marcotte); and

an operating system kernel operatively connected to the file system (see col. 18, lines 50-53, Marcotte), the operating system kernel operative to enable task execution for at least one application (see Abstract), the operating system kernel further comprising:

a file system namespace cache for caching file paths from the file system namespace (see col. 18, lines 26-33, Marcotte); and

a file look-up operation operable to determine (see col. 18, lines 54-58, Fig. 3, Marcotte) if a specific file path corresponding to a specific file is stored in the file system namespace cache and to selectively enable performance of a file operation relative to the specific file without suspending task execution(see col. 14, lines 5-13, Marcotte) and notify the at least one application when the a file operation relative to the specific file is not being performed suspending task execution based on the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) not being performed without suspending task execution (see col. 14, lines 5-13, Marcotte) because the specific file path is not stored in the file system namespace cache so that the at least one application can redirect a request to perform the file operation (see col. 6, lines 47-55, Marcotte).

Marcotte does not explicitly indicate the claimed namespace for cache.

Pike discloses claimed namespace for cache (an operating system root service which provides a predefined file tree for binding to the root of the prcess's name space, see col. 3, lines 59-61, Pike).

It would have been obvious to one ordinary skill in the data processing art, at the time of the present invention to combine teachings of the cited references because the namespace for cache of Pike's teachings would have allowed Marcotte's system for multiprocess operating system which characterized in different name, as suggested by Pike at col. 3, lines 41-45. Further, Namespace for cache as taught by Pike improves to files control entity on per-process basis (see col. 3, lines 64-65, Pike).

As to claim 16,

Marcotte teaches wherein the at least one application resides and executes within a user space that is operatively connected to the operating system kernel and the file system namespace (see col. 18, lines 50-53, Marcotte).

As to claim 17,

Marcotte teaches wherein the at least one application resides and executes within the operating system kernel (see col. 18, lines 50-53, Marcotte).

As to claim 18,

Marcotte teaches wherein the operating system kernel further comprises blocking point handling (see col. 18, lines 50-53, Marcotte) which can be invoked if and when the file operation cannot be performed-atomically without suspending task execution (see col. 14, lines 5-13, Marcotte).

As to claim 19,

Marcotte teaches wherein the operating system kernel further comprises blocking point handling which can be invoked if and when the file operation cannot be performed atemicall-y without suspending task execution (see col. 14, lines 5-13, Marcotte).

As to claim 20,

Marcotte teaches wherein the operating svstem kcr iici f ui thur comprises blocking point handling which can be invoked if and when the fiie operation cannot be performed without suspending task execution (see col. 14, lines 5-13, Marcotte).

As to claim 21,

Marcotte teaches wherein the user space further comprises blocking point handling which can be invoked if and when the file operation cannot be performed without suspending task execution (see col. 14, lines 5-13, Marcotte).

As to claim 22,

Marcotte teaches wherein the operating system kernel and the user space further comprise: blocking point handling which can be invoked if and when the file operation cannot be performed without suspending task execution (see col. 14, lines 5-13, Marcotte).

With respect to claim 23,

Marcotte discloses an instruction execution system operable to handle a request from an application to an operating system to perform a file operation relative to a specific file by performing (see col. 13, lines 50-60, Marcotte) the steps of:

sending the request to perform the file operation (see col. 16, lines 8-14, Marcotte) from the application to the operating system wherein the request causes the op -rating system to perform a file lookup operation (see col. 18, lines 54-58, Fig. 3, Marcotte);

attempting to perform the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) without suspending task execution by (see col. 14, lines 5-13, Marcotte), at least in part, determining if a file path corresponding to the specific file is stored in a file system namespace cache (see col. 18, lines 26-33, Marcotte);

notifying the application that the file operation could not be performed without suspending task execution (see col. 14, lines 5-13, Marcotte) so that the application can appropriately redirect the request if the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) could not be performed without suspending task execution because the file path is not stored in the file system namespace cache (see col. 18, lines 26-33, Marcotte);

redirecting the request (see col. 6, lines 47-55, Marcotte) if the.file operation was not performed without suspending task execution (see col. 14, lines 5-13, Marcotte) because the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) could not be performed without suspending task execution (see col. 14, lines 5-13, Marcotte).

Marcotte does not explicitly indicate the claimed namespace for cache.

Pike discloses claimed namespace for cache (an operating system root service which provides a predefined file tree for binding to the root of the prcess's name space, see col. 3, lines 59-61, Pike).

It would have been obvious to one ordinary skill in the data processing art, at the time of the present invention to combine teachings of the cited references because the namespace for cache of Pike's teachings would have allowed Marcotte's system for multiprocess operating system which characterized in different name, as suggested by Pike at col. 3, lines 41-45. Further, Namespace for cache as taught by Pike improves to files control entity on per-process basis (see col. 3, lines 64-65, Pike).

As to claim 24,

Marcotte teaches wherein the file system namespace cache is disposed within an operating system kernel (see col. 18, lines 50-53, Marcotte).

As to claim 25,

Marcotte teaches wherein the redirecting of the request further comprises sending the request to blocking point handling within a user space including the application (see col. 6, lines 47-55 et seq, Marcotte).

As to claim 26,

Marcotte teaches wherein the redirecting of the request further comprises sending the request to blocking point handling within a kernel of the operating system (see col. 18, lines 50-53, Marcotte).

As to claim 27,

Marcotte teaches wherein the redirecting of the request further comprises sending the request to blocking point handling within a user space including the application (see col. 6, lines 47-55 et seq, Marcotte).

As to claim 28,

Marcotte teaches wherein the redirecting of the request further comprises sending the request to blocking point handling within a kernel of the operating system (see col. 18, lines 50-53, Marcotte).

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in a patent granted on an application for patent by another filed in the United States before the invention thereof by the applicant for patent, or on an international application by another who has fulfilled the requirements of paragraphs (1), (2), and (4) of section 371(c) of this title before the invention thereof by the applicant for patent.

The changes made to 35 U.S.C. 102(e) by the American Inventors Protection Act of 1999 (AIPA) and the Intellectual Property and High Technology Technical Amendments Act of 2002 do not apply when the reference is a U.S. patent resulting directly or indirectly from an international application filed before November 29, 2000. Therefore, the prior art date of the reference is determined under 35 U.S.C. 102(e) prior to the amendment by the AIPA (pre-AIPA 35 U.S.C. 102(e)).

5. Claims 5-7, 10-12, and 14 are rejected under 35 U.S.C. 102(e) as being anticipated by Scott T. Marcotte ('Marcotte' hereinafter), US Patent 6,449,614.

With respect to claim 5,

Marcotte discloses a method of handling a request to an operating system to perform a file operation, the request being sent from an application to the operating system, wherein the operating system can notify the application if the file operation cannot be performed without suspending task execution (see col. 13, lines 50-60, Marcotte), the method comprising the steps of:

sending the request to the operating system (see col. 16, lines 8-14, Marcotte), wherein the request causes the operating system to perform a file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte);

receiving a notification from the operating system that the file operation cannot be performed without suspending task execution (see col. 14, lines 5-13, Marcotte) if the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) cannot be performed without suspending task execution (see col. 14, lines 5-13, Marcotte), the notification enabling the request to be redirected (see col. 18, lines 26-33, Marcotte); and

redirecting the request to blocking point handling (see col. 6, lines 47-55, Marcotte) if the notification is received from the operating system that the file operation was not performed without suspending task execution (see col. 14, lines 5-13, Marcotte) because the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) could not be performed without suspending task execution (see col. 14, lines 5-13, Marcotte).

As to claim 6,

Marcotte teaches wherein the blocking point handling resides within a user space including the application (see col. 6, lines 47-55, Marcotte).

As to claim 7,

Marcotte teaches wherein blocking point handling resides within a kernal for the operating system (see col. 6, lines 47-55, Marcotte).

With respect to claim 10,

Marcotte discloses computer program product having computer program code embodied therein, the computer program code for handling a request to an operating system to perform a file operation relative to a specific file, the computer program code (see col. 13, lines 50-60, Marcotte) comprising:

instructions for sending the request to the operating system (see col. 16, lines 8-14, Marcotte), wherein the request causes the operating system to perform a file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte);

instructions for receiving a notification from the operating system that the file operation cannot be performed atomically without suspending task execution (see col. 14, lines 5-13, Marcotte) if the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) cannot be performed without suspending task execution (see col. 14, lines 5-13, Marcotte), the notification enabling the request to be redirected (see col. 18, lines 26-33, Marcotte);

instructions for using the specific file if the file operation was performed without suspending task execution (see col. 14, lines 5-13, Marcotte) because the file lookup operation was performed without suspending task execution (see col. 14, lines 5-13, Marcotte); and

instructions for redirecting the request to blocking point handling if the notification is received from the operating system that the file operation was not performed without suspending task execution (see col. 14, lines 5-13, Marcotte) because the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) could not be performed without suspending task execution (see col. 14, lines 5-13, Marcotte).

As to claim 11,

Marcotte teaches wherein blocking point handling resides within a user space including the application.

As to claim 12,

Marcotte teaches wherein the blocking point handling resides within the an operating system kernel (see col. 18, lines 50-53, Marcotte).

With respect to claim 14,

Marcotte discloses apparatus for handling a request to an operating system to perform a file operation relative to a specific file (see col. 13, lines 50-60, Marcotte), the apparatus comprising:

means for sending the request to the operating, (see col. 16, lines 8-14, Marcotte) system wherein the request causes the operating system to perform a file look-up operation;

means for receiving a notification from the operating system that the file operation cannot be performed without suspending task execution (see col. 14, lines 5-13, Marcotte) if the file look-up operation cannot be performed without suspending task execution, the notification enabling the request to be redirected (see col. 14, lines 5-13, Marcotte);

means for using the specific file if the file operation was performed received without suspending task execution because (see col. 14, lines 5-13, Marcotte) the file look up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) was performed without suspendin task execution (see col. 18, lines 26-33, Marcotte); and

means for redirecting the request to blocking Joint handling if the notification is received from the operating system that the file operation was not performed without suspending task execution because the file look-up operation (see col. 18, lines 54-58, Fig. 3, Marcotte) could not be performed without suspending task execution (see col. 18, lines 26-33, Marcotte).

Contact Information

6. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mohammad Ali whose telephone number is (703) 605-4356. The examiner can normally be reached on Monday to Thursday from 7:30am-4:30pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John Breene can be reached on (703) 305-9790 or TC 2100 customer service (703) 306-5631. The fax phone numbers for the organization where this application or proceeding is assigned are (703) 872-9306 for regular communications.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-9600.

MA
June 9, 2004

Mohammad Ali
Patent Examiner
Art Unit : 2177